(19) BUNDESREPUBLIK DEUTSCHLAND

DEUTSCHES PATENTAMT

(12) **Offenlegungsschrift**

(10) **DE 195 36 548 A 1**

(51) Int. Cl.[6]:
G 06 F 9/45
G 06 K 1/00

(21) Aktenzeichen: 195 36 548.8
(22) Anmeldetag: 29. 9. 95
(43) Offenlegungstag: 3. 4. 97

DE 195 36 548 A 1

(71) Anmelder:
International Business Machines Corp., Armonk, N.Y., US

(74) Vertreter:
Kauffmann, W., Dipl.Phys. Dr., Pat.-Ass., 70569 Stuttgart

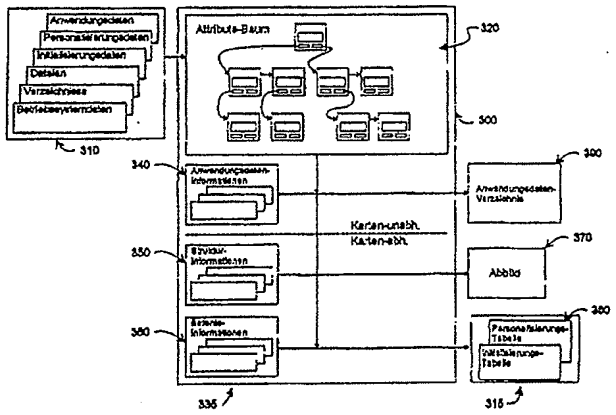
(72) Erfinder:
Bublitz, Hermann, 71034 Böblingen, DE; Rindtroff, Klaus, 71093 Weil im Schönbuch, DE

(56) Entgegenhaltungen:
DE 38 35 479 C2
US 53 94 546

Prüfungsantrag gem. § 44 PatG ist gestellt

(54) Vorrichtung und Verfahren zur vereinfachten Erzeugung von Werkzeugen zur Initialisierung und Personalisierung von und zur Kommunikation mit einer Chipkarte

(57) Vorgestellt wird ein Übersetzer (300) zur Erstellung eines Werkzeuges (315) für eine Modifikation von Daten auf einer Chipkarte (10) oder für eine Kommunikation mit der Chipkarte (10), wobei der Übersetzer (300) als Eingangsdaten eine Beschreibung (310) in einer hierfür vorgesehenen Kartendefinitionssprache erhält und die Beschreibung (310) Information über Objekte auf der Chipkarte (10), die mit Hilfe konstruktiver Anweisungen beschrieben sind, aufweist. Der Übersetzer (300) enthält ein Mittel zur Generierung eines internen Attributeverzeichnisses (320) aus den Informationen über die Objekte in der Beschreibung (310), wobei für jedes Objekt, mindestens ein Eintrag (330) in dem internen Attributeverzeichnis (320) existiert. Weiterhin enthält der Übersetzer (300) zumindest ein Strukturinformationsmittel (335), das beschreibt, wie die in dem mindestens einen Eintrag (330) in der internen Attributeverzeichnis (320) gespeicherten Daten in das für die Chipkarte (10) bestimmte Format gebracht werden, welche Informationen aus dem mindestens einen Eintrag (330) verwendet und in welchem Format sie in der speziellen Struktur abgelegt werden.



DE 195 36 548 A 1

^CID: <DE_____19536548A1_I_>

Beschreibung

Gebiet der Erfindung

Die Erfindung betrifft die Erzeugung von Werkzeugen zur Initialisierung und Personalisierung von und zur Kommunikation mit einer Chipkarte.

Stand der Technik

Als Datenträgerkarten oder Chipkarten werden heute tragbare Karten, zumeist in etwa in Scheckkartenformat und vorzugsweise aus Kunststoff oder Metall, mit einem darin integrierten elektronischen Chip, bezeichnet. Man unterscheidet zwischen einfachen Speicherkarten (die als Memory Chip-Cards oder Memory-Cards bekannt sind) und intelligenten Datenträgerkarten (auch multifunktionale Chipkarten, oder Smart-Cards genannt), die neben einem Speicherbereich einen eigenen Prozessor zur Kontrolle der auf dem Chip der Datenträgerkarte gespeicherten Daten aufweisen. Dies erlaubt einen besseren Schutz der Daten und führt zu einer verbesserten Funktionalität der Chipkarten. Einfache Speicherkarten erlauben im allgemeinen nur ein Schreiben und Lesen der Daten. Intelligente Datenträgerkarten verfügen darüber hinaus noch über Funktionen zur Strukturierung der Daten, zur Lokalisierung der Daten, zur Verwaltung der Daten und zum Schutz der Daten. Die Schnittstelle einer intelligenten Chipkarte, und damit auch die erforderliche Programmierung (z. B. von Kommandosequenzen), um Daten von einer Chipkarte zu lesen, ist daher wesentlich komplexer als die von Speicherkarten.

Fig. 1 zeigt die verschiedenen Phasen einer Chipkarte **10**. Eine erste Phase **0** stellt den Herstellungsprozeß der Chipkarte **10** dar. Am Ende des Herstellungsprozesses liegt die Chipkarte **10** in einem noch nicht gebrauchsfähigen Zustand vor. Die Chipkarte **10** enthält i.a. bis auf ein fest eingeschriebenes Betriebssystem keine Daten, und die beschreibbaren Speicher liegen in einem leeren Zustand vor.

Um die Chipkarte **10** in einen gebrauchsfähigen Zustand zu überführen, wird diese in einer Phase I initialisiert, das heißt, die logische Kartenstruktur (siehe Fig. 2) der Chipkarte **10** wird durch einen Initialisierungsprozeß auf der Chipkarte **10** festgelegt. Im Speicher der Chipkarte wird ein Dateiensystem angelegt und es werden zusätzliche Daten für das Betriebssystem, wie z. B. Kommunikationsparameter und sicherheitsbedingte Parameter (wie Schlüsselwörter oder Paßwörter) abgelegt.

Fig. 2 zeigt den typischen Aufbau einer logischen Struktur in einer Chipkarte **10**. Die Struktur eines Systems beschreibt ihre statischen Eigenschaften, das heißt, welche Komponenten auf welche Art miteinander verknüpft sind. Die logische Struktur erfolgt dabei nach inhaltlichen Gesichtspunkten und bildet vorzugsweise eine Baumstruktur. Die Strukturelemente der logischen Struktur werden auch logische Objekte genannt. Die Wurzel des logischen Strukturbaums stellt ein Wurzelverzeichnis **100** dar. An das Wurzelverzeichnis **100** geknüpft, können eine Vielzahl von zusammengesetzten logischen Objekten oder eine Vielzahl elementarer logischer Objekte sein. Beispiel eines elementaren logischen Objektes in Fig. 2 ist eine Datei **110**, die direkt mit dem Wurzelverzeichnis **100** verknüpft ist und keine weiteren Verweise mehr aufweist. Fig. 2 zeigt weiterhin eine Reihe zusammengesetzter logischer Objekte, wie ein Unterverzeichnis **120**, das mit 2 Dateien **130** und **140** verbunden ist und ein weiteres Unterverzeichnis **150**, das wiederum mit 2 weiteren Unterverzeichnissen **160** und **170** verknüpft ist, wobei das Unterverzeichnis **160** eine Datei **180** beinhaltet und das Unterverzeichnis **170** ein weiteres Unterverzeichnis **190** usw. aufweist.

Jedes Verzeichnis oder Unterverzeichnis stellt eine Abbildung von Namen auf die dem jeweiligen Verzeichnis zugeordneten Objekte dar. Jedes Objekt der logischen Struktur, also jedes einzelne der Elemente **100** bis **190** wie in Fig. 2 gezeigt, weist eine Reihe von Attributen, wie z. B. einen Identifikationsnamen, Paßwortschutz, Zugriffsweise usw., auf.

Die logische Kartenstruktur ist im Bezug auf die spätere Verwendung der Chipkarte **10**, z. B. durch einen Anwendungsprogrammierer, vorzugeben. Es sind die logische Dateienstruktur und die Verknüpfung der unterschiedlichen Dateien anzugeben. Den jeweiligen Dateien können verschiedene Attribute zugeordnet werden, wie Paßwortschutz, nur Lese- oder nur Schreibzugriff, Lese- und Schreibzugriff, usw., die in einer Dateibeschreibung spezifiziert werden. Für den Zugang zu den Datensätzen einer Datei werden im allgemeinen Ordnungsbegriffe vergeben, die eine Adressierung jedes einzelnen Datensatzes der Datei ermöglichen. Bei der Festlegung der logischen Kartenstruktur einer Chipkarte **10** ist zu berücksichtigen, daß die Semantik der Identifikationsnamen und Attribute zum Teil durch Standards oder durch die einzelnen Kartenanbieter festgelegt ist.

Der Initialisierungsvorgang wird für alle Chipkarten **10** eines Typs und für eine spätere Verwendungsweise der Chipkarte **10** gleich durchgeführt. Bei der Initialisierung der Chipkarte **10** in der Phase I ist zwischen einer Masseninitialisierung MI und einer Einzelinitialisierung EI zu unterscheiden (Fig. 1). Die Masseninitialisierung MI erfolgt i.a. in einem großen industriellen Maßstab mit speziell dafür ausgerichteten Maschinen. Die Einzelinitialisierung EI hingegen wird nur mit kleineren Stückzahlen durchgeführt und dient in erster Linie zur Herstellung von Labor- oder Musterstücken, sowie zur Anfertigung kleinerer individueller Serien. Im Gegensatz zur Masseninitialisierung erfolgt die Einzelinitialisierung in der Regel mit nicht spezialisierten Apparaten, die für diese Einzelinitialisierung speziell eingerichtet werden müssen.

Der Einzelinitialisierungsprozeß EI der Chipkarte **10** wird im allgemeinen so durchgeführt, daß ein binäres Initialisierungs-Bild **200** eines zu beschreibenden Speichers (z. B. eines EEPROM oder eines optischen Speichers) der Chipkarte **10** von einem, mit den internen Details der Chipkarte **10** eines jeweiligen Chipkartentypes vertrauten, Chipkartenprogrammierers erstellt wird. Dabei stellt das binäre Initialisierungs-Bild **200** das Spiegelbild des zu beschreibenden Speichers nach der Initialisierung dar, oder anders ausgedrückt, enthält das binäre Initialisierungs-Bild **200** die Informationen, wie der zu beschreibende Speicher beschrieben werden soll. Das

binäre Initialisierungs-Bild **200** wird mit Hilfe eines hierfür zu erstellenden Programmes in den zu beschreibenden Speicher "kopiert". Dabei erstellt das Programm die erforderlichen Anweisungen für den "Kopiervorgang".

Analog dazu erfolgt der Masseninitialisierungsprozeß MI der Chipkarte **10** im allgemeinen mit Hilfe einer Initialisierungstabelle **210**. Die Initialisierungstabelle **210** enthält die komplette Initialisierungsvorschrift mit den für die Initialisierung erforderlichen Kommandos und binären Daten. Die Maschine, die die Masseninitialisierung durchführt, muß den jeweiligen Chipkartentyp nicht kennen, sondern kann mit Hilfe der Initialisierungstabelle **210** die Chipkarte **10** unmittelbar initialisieren. Dies ist insbesondere bei der Initialisierung von großen Stückzahlen, evtl. auch unterschiedlicher Chipkartentypen, von Vorteil. Die Initialisierungstabelle **210** wird zumeist durch Umwandlung des binären Initialisierungs-Bildes **200** erstellt.

Am Ende des Initialisierungsprozesses erhält man eine betriebsbereite Chipkarte **10** mit einer festgelegten logischen Karten- und Dateienstruktur. Alle so mit einem gleichen Initialisierungsverfahren initialisierten Chipkarten **10** eines Chipkartentypes sind identisch zueinander.

Um die Chipkarte **10** zu individualisieren und beispielsweise mit Daten des Karteninhabers zu versorgen, werden die Chipkarten **10** in einer weiteren Phase P personalisiert. Die Personalisierung erfolgt im allgemeinen individuell für jede einzelne Chipkarte **10**. Durch den Personalisierungsvorgang wird die Chipkarte **10** mit persönlichen Daten des Karteninhabers versorgt. Hierbei ist zwischen einer Massenpersonalisierung MP und einer Einzelpersonalisierung EP zu unterscheiden. Bei der Massenpersonalisierung MP erfolgt der Personalisierungsvorgang in erster Linie zentral mit Hilfe in Datenbanken gespeicherter Daten, z. B. einer Kundendatei oder dergleichen. Die Einzelpersonalisierung EP hingegen erfolgt im allgemeinen dadurch, daß einzelne Karten speziell und in einem Einzelprozeß dezentral mit Daten des jeweiligen Karteninhabers versorgt werden.

Sowohl die Massenpersonalisierung MP als auch die Einzelpersonalisierung EP der Chipkarte **10** wird im allgemeinen mit Hilfe einer Personalisierungstabelle **220** durchgeführt. Die Personalisierungstabelle **220** enthält die erforderlichen Anweisungen zur Personalisierung der Chipkarte **10** und Referenzen auf Daten in Personalisierungsdateien (z. B. einer Kundendatei, Mitgliederdateien, usw.). Die Personalisierungstabelle **220** wird durch ein Programm interpretiert, das die individuellen Daten einsetzt. Die Anweisungen werden direkt an die Chipkarte **10** gerichtet und von dieser umgesetzt. Die Personalisierungstabelle **220** enthält man durch Umsetzung eines binären Abbildes (z. B. eines binären Personalisierungsbildes **230**) und der Referenzen. Entsprechend der Einzelinitialisierung EI kann die Einzelpersonalisierung EP auch unmittelbar mit Hilfe des Personalisierungsbildes **230** durchgeführt werden.

Am Ende der Personalisierungsphase P ist die Chipkarte **10** in einem betriebsbereiten Zustand versetzt und sollte, nach Möglichkeit, die für die geplanten Anwendungen der Chipkarte **10** erforderlichen Daten gespeichert haben. Während der Initialisierungs- und/oder Personalisierungsphase können die einzelnen Datensätze der Dateien bereits mit Startwerten versehen werden.

Die erfolgreich initialisierte und personalisierte Chipkarte **10** kann nun an den Karteninhaber ausgegeben werden und von diesem in einer Anwendungs-Phase A für die mit der Chipkarte **10** vorgesehenen Anwendungen eingesetzt werden.

Es ist zu verstehen, daß die Initialisierungs- und Personalisierungsvorgänge im allgemeinen irreversible Vorgänge sind, d. h. daß die Chipkarte **10** aus Sicherheitsgründen vor einer Initialisierung vollständig gelöscht wird. Auch kann die Chipkarte **10** nach einer durchgeführten Personalisierung so verändert werden, daß eine weitere Personalisierung nicht mehr erlaubt ist. Genauso lassen sich jedoch auch Zustände definieren, die eine Neu-Personalisierung oder eine Teil-Personalisierung erlauben. Insbesondere bei einer dezentralen Personalisierung, wie der Einzelpersonalisierung, können z. B. aus der Anwendungsphase heraus auch einzelne Bereiche der Chipkarte **10** wieder neu initialisiert oder personalisiert werden. Ebenso ist es möglich, neue Datenstrukturen anzulegen und zu personalisieren.

Anwendungen mit Chipkarten **10** können beispielsweise bargeldloses Bezahlen, Identifikation des Chipkarteninhabers, Speicherung von Daten oder dergleichen sein. Eine Anwendung besteht dabei aus internen Anwendungsteilen auf der Chipkarte **10** und externen Anwendungsteilen in entsprechenden Geräten, wie z. B. Geldautomaten, PC's oder speziellen Terminals. Allgemein stellen interne Anwendungsteile alle Daten und Programme dar, die auf der Chipkarte **10** selbst gespeichert werden, während die externen Anwendungsteile alle Daten und Programme außerhalb der Chipkarte **10** darstellen.

Eine Kommunikation mit der Chipkarte **10** in der Anwendungsphase A geschieht i.a. durch die Verwendung von Programmierschnittstellen. Aus der deutschen Patentanmeldung mit dem Aktenzeichen P 195 22 527.9 [GE 9 95 014] der Anmelderin ist eine solche Programmierschnittstelle bekannt. Die Lehre dieser Anmeldung in Bezug auf die Schnittstelle zur Chipkarte und die Kommunikation mit der Chipkarte, sowie deren Implementierungen ist Bestandteil der vorliegenden Erfindung. Gemäß der Patentanmeldung P 195 22 527.9 erfolgt eine Trennung zwischen anwendungsspezifischen Daten und kartenspezifischen Daten. Anwendungsspezifische Daten sind dabei solche Daten, die Informationen über Art, Lokalität, Umfang und Zugriffsmethoden für auf einer Chipkarte gespeicherte Daten enthalten, sowie die auf der Chipkarte gespeicherten Daten selbst. Kartenspezifische Daten dagegen stellen solche Daten dar, die Informationen über die erforderlichen Kommandos und das Protokoll der Chipkarte zum Zugriff auf die dort gespeicherten Daten geben. Diese Trennung zwischen anwendungsspezifischen Daten und kartenspezifischen Daten ermöglicht es, daß ein und dieselbe Anwendung mit verschiedenen Chipkartentypen realisiert werden kann. Dies führt zu einer wesentlichen Vereinfachung der Schnittstelle zu Chipkarten und verbessert andererseits die Trennung zwischen internen Anwendungsteilen auf der Chipkarte und von der Chipkarte getrennten, externen Anwendungsteilen. Eine flexible Anpassung an neue Anwendungen und Chipkartentypen wird so unterstützt.

Weiter in **Fig. 1**, dient für eine Kommunikation mit der Chipkarte **10** in der Anwendungsphase A beispielsweise ein Anwendungsdatenverzeichnis gemäß der oben genannten Patentanmeldung P 195 22 527.9, das sogenannte Wörterbuch **250** (application dictionary), zur Aufnahme von anwendungsspezifischen Daten. Ein Chipkarten-

dialogmodul 260, der sogenannte Agent 260 (smart card agent oder smart card interpreter), generiert die erforderlichen Kommandos für die Schnittstelle der Chipkarte 10. Das Wörterbuch 250 enthält dabei Informationen über Art, Lokalität, Umfang und Zugriffsmethoden von auf der Chipkarte 10 gespeicherten Daten, sowie Informationen für die, aufgrund der Sicherheitsvorgaben der Anwendung eventuell erforderliche, Behandlung dieser Daten. Zur einfachen Identifizierung der jeweiligen Daten für einen Zugriff durch die Anwendung werden den Daten ein oder mehrere Alias-Namen zugeordnet. Diese Informationen werden in geeigneter Form, wie z. B. tabellarisch oder hierarchisch, in dem Wörterbuch 250 angelegt und umfassen alle erforderlichen Informationen für die Anwendung oder eine Vielzahl weiterer Anwendungen. Der Zugriff auf das Wörterbuch 250 erfolgt hierbei durch den Agenten 260. Die Zuordnung eines entsprechenden Wörterbuchs 250 zu einem für eine Anwendung entsprechenden Agenten 260 kommt durch den externen Anwendungsteil oder einer speziellen Erweiterung des Agenten 260 zustande.

Um die Initialisierungs- und Personalisierungsprozesse der Chipkarte 10 durchführen zu können, sind, genauso wie bei einer späteren Kommunikation einer Anwendung mit der Chipkarte 10, detaillierte Kenntnisse über die internen Strukturen und Abläufe, sowie über die kartenspezifischen Kommandos und Standardisierungen notwendig. Weiterhin müssen detaillierte Kenntnisse über die Anwendungen sowie deren Implementierungen und die Kommunikation mit der Chipkarte 10 vorliegen. Da häufig jedoch die Personenkreise, die sich entweder mit den Anwendungen oder mit der Chipkarte 10 beschäftigen nicht identisch sind, ist so ein hohes Spezialistentum erforderlich, das detaillierte Kenntnisse sowohl über die Chipkarte 10, die jeweiligen Anwendungen der Chipkarte, als auch über den Initialisierungs- und den Personalisierungsprozeß aufweisen muß.

Es ist deshalb Aufgabe der Erfindung, die Bereitstellung funktionsfähiger Chipkarten sowie die Umsetzung von Anwendungen mit Chipkarten zu erleichtern. Die Aufgabe der Erfindung wird entsprechend der unabhängigen Ansprüche gelöst. Weitere, vorteilhafte Ausführungen der Erfindung finden sich in den Unteransprüchen.

Die Erfindung weist auf einen Übersetzer zur Erstellung eines Werkzeuges für eine Modifikation von Daten auf einer Chipkarte oder für eine Kommunikation mit der Chipkarte, wobei der Übersetzer als Eingangsdaten eine Beschreibung in einer hierfür vorgesehenen Kartendefinitionssprache erhält und die Beschreibung Information über Objekte auf der Chipkarte, die mit Hilfe konstruktiver Anweisungen beschrieben sind, aufweist. Der Übersetzer enthält ein Mittel zur Generierung eines internen Attributeverzeichnisses aus den Informationen über die Objekte in der Beschreibung, wobei für jedes Objekt, mindestens ein Eintrag in dem internen Attributeverzeichnis existiert. Weiterhin enthält der Übersetzer zumindest ein Strukturinformationsmittel, das beschreibt, wie die in dem mindestens einen Eintrag in der internen Attributeverzeichnis gespeicherten Daten in das für die Chipkarte bestimmte Format gebracht werden, welche Informationen aus dem mindestens einen Eintrag verwendet und in welchem Format sie in der speziellen Struktur abgelegt werden.

Erfindungsgemäß wird ein Verfahren durchgeführt, bestehend aus: Eingabe der Beschreibung der Chipkarte, Generierung des internen Attributeverzeichnisses aus den Informationen über die Objekte in der Beschreibung, Eingabe und/oder Abruf des zumindest einen Strukturinformationsmittels, und Erzeugung des Werkzeuges aus der Beschreibung mit Hilfe des internen Attributeverzeichnisses und des zumindest einen Strukturinformationsmittels.

Beschreibung der Zeichnungen

Zur näheren Erläuterung der Erfindung sind im folgenden Ausführungsbeispiele mit Bezugnahme auf die Zeichnungen beschrieben.

Fig. 1 zeigt die verschiedenen Phasen einer Chipkarte vom Herstellungsprozeß über die Initialisierung, die Personalisierung bis zur Anwendung;

Fig. 2 zeigt den typischen Aufbau einer logischen Struktur in einer Chipkarte;

Fig. 3 zeigt den Aufbau eines erfindungsgemäßen Übersetzers zur vereinfachten Erzeugung von Hilfsmitteln in den verschiedenen Phasen einer Chipkarte;

Fig. 4 zeigt einen Attribute-Baum;

Fig. 5 zeigt einen Knoten in dem Attribute-Baum; und

Fig. 6 zeigt die Reihenfolge der Verarbeitung der Abbilder für den zu beschreibenden Speicher der Chipkarte.

Allgemeine Beschreibung der Erfindung

Fig. 3 zeigt den Aufbau eines erfindungsgemäßen Übersetzers 300 zur vereinfachten Erzeugung von Werkzeugen, wahlweise für die Initialisierung und/oder Personalisierung der Chipkarte 10 und/oder für die Kommunikation mit der betriebsbereiten Chipkarte 10. Der Übersetzer 300 erhält als Eingang (Source) eine Beschreibung 310 der Chipkarte 10 in einer hierfür vorgesehenen Kartendefinitionssprache. Die Beschreibung 310 enthält Information über Objekte auf der Chipkarte 10, die mit Hilfe konstruktiver Anweisungen beschrieben sind. Vorzugsweise enthält die Beschreibung 310 Information über:

- Betriebssystemdaten,
- Verzeichnisse,
- Dateien,
- Initialisierungsdaten und
- Personalisierungsdaten

der Chipkarte 10, sowie

— Anwendungsdaten

für eine Kommunikation mit der Chipkarte 10.

Die Beschreibung 310 weist Attribute-Anweisungen auf, die die Eigenschaften der Objekte spezifizieren (nähere Erläuterung hierzu unter Kartendefinitionssprache).

Die auf der Chipkarte 10 erreichbaren Daten sind in der Beschreibung 310 vorzugsweise mit dafür vorgesehenen Alias-Namen versehen. Die Beschreibung 310 liegt hierfür in einer für Menschen lesbaren Form und im Syntax einer speziellen Beschreibungssprache, der Kartendefinitionssprache, für den Übersetzer 300 vor. Der Übersetzer 300 erzeugt aus der Beschreibung 310 die Daten und Kommandos, die, je nach Bedarf, für eine entsprechende Anwendung wie Initialisierung, Personalisierung oder Kommunikation mit der Chipkarte 10 erforderlich sind.

Der Übersetzer 300 erzeugt aus der Beschreibung 310 spezielle Strukturen 315 für eine Umsetzung von Daten (z. B. bei der Initialisierung oder der Personalisierung) auf die Chipkarte 10 und für die Kommunikation mit der Chipkarte 10. Die speziellen Strukturen 315 ermöglichen eine Umsetzung der, der Beschreibung 310 entsprechenden gewünschten Struktur, in eine physikalische Struktur auf der Chipkarte 10. Die Umsetzung kann dabei entweder durch das unmittelbare Zurverfügungstellen eines entsprechenden Werkzeuges oder durch das mittelbare Erzeugen von Hilfsmitteln für die Erstellung der entsprechenden Werkzeuge geschehen.

Der Übersetzer 300 generiert aus den Informationen über die Objekte in der Beschreibung 310 eine interne Baumstruktur 320 (siehe dazu auch Fig. 4), den sogenannten Attribute-Baum 320. Für jedes Objekt, definiert durch eine konstruktive Anweisung, existiert ein Knoten 330 in dem Baum der internen Baumstruktur 320. Jeder Knoten 330 (Fig. 5) enthält:

— einen Namen des Objektes,
— einen Typ des Objektes,
— Attribute des Objektes, wie in den Attribute-Anweisungen spezifiziert, und
— Verknüpfungen mit anderen Knoten im Attribute-Baum 320.

Der Attribute-Baum 320 ist unabhängig von dem jeweiligen, verwendeten Chipkartentyp. Es ist zu verstehen, daß der Attribute-Baum 320 allgemein ein Attributeverzeichnis darstellt, das als ein sequentielles hierarchisches Verzeichnis ausgeprägt ist und vorzugsweise eine Baumstruktur bildet.

Der Übersetzer 300 enthält allgemeine Strukturinformationsmittel 335 (sogenannte Templates), wie Anwendungsdaten-Informationen 340, Struktur-Informationen 350 und Befehls-Informationen 360. Der Übersetzer 300 erzeugt hieraus wahlweise beispielsweise ein Abbild 370 des zu beschreibenden Speichers der Chipkarte 10, eine Tabelle 380 oder ein Anwendungsdatenverzeichnis 390.

Die Struktur-Informationen 350 und Befehls-Informationen 360 beschreiben wie die in den Knoten des Attribute-Baums 320 gespeicherten Daten in ein für die Chipkarte bestimmtes Format gebracht werden sollen. Sie bestimmen welche Informationen aus den Knoten verwendet werden und in welchem Format sie in den speziellen Strukturen 315 abgelegt werden. Um die speziellen Strukturen 315 für eine beliebige Chipkarte zu generieren, genügt es also die allgemeinen Strukturinformationsmittel 335 an die Eigenschaften der Chipkarte anzupassen.

Das Abbild 370 des zu beschreibenden Speichers (vorzugsweise ein EEPROM) der Chipkarte 10 stellt vorzugsweise das binäre Bild 200 für die Einzelinitialisierung EI oder das Personalisierungsbild 230 dar.

Die Tabelle 380 kann beispielweise die Initialisierungstabelle 210 für die Masseninitialisierung MI oder die Personalisierungstabelle 220 für die Massenpersonalisierung MP der Chipkarte 10 sein.

Das Anwendungsdatenverzeichnis 390 enthält anwendungsspezifische Daten die für eine Kommunikation mit der Chipkarte erforderlich sind. Das Anwendungsdatenverzeichnis 390 stellt hierbei vorzugsweise das Wörterbuch 250 (siehe in Fig. 1), das die anwendungsspezifischen Daten für eine Kommunikation mit der Chipkarte 10 z. B. durch den Agenten 260 gemäß der oben genannten Patentanmeldung P 195 22 527.9 enthält, dar.

Es ist zu verstehen, daß der Übersetzer 300 auch weitere als die genannten Werkzeuge für die Vorbereitung und Bereitstellung der Chipkarte 10 sowie für die Kommunikation mit der Chipkarte 10 zur Verfügung stellen kann. Diese Werkzeuge werden dann durch die geforderte Anwendung der Chipkarte 10 definiert.

Die Anwendungsdaten-Informationen 340 weisen vorzugsweise ein allgemeines Layout des Anwendungsdatenverzeichnisses 390 auf.

Die Struktur-Informationen 350 weisen vorzugsweise Informationen über

— das Verzeichnis als allgemeines Layout der internen physischen Struktur auf der Chipkarte 10 (zum Beispiel: Der interne Datensatz ist 16 Bytes lang, Byte 1 ist immer 0 × 63, Byte 2 gibt die Länge der Daten an, usw.);
— die Dateien als allgemeines Layout der internen physischen Struktur auf der Chipkarte 10; und
— die Betriebssystemdaten als allgemeines Layout der internen Betriebssystemstruktur;

im entsprechenden Format ("Bits und Bytes") der Chipkarte 10 auf.

Die Befehls-Informationen 360 weisen vorzugsweise Informationen über

— die Initialisierungstabelle 210 und
— die Personalisierungstabelle 220,

als allgemeines Layout der jeweiligen Tabelle und der entsprechenden erforderlichen Kommandos, auf.

Die Struktur-Information 340 für das Anwendungsdatenverzeichnis 390 ist von dem Typ der Chipkarte 10 unabhängig. Die Initialisierungstabelle 210, die Personalisierungstabelle 220 und das Abbild 370 sind von dem Typ der Chipkarte 10 abhängig, das heißt, daß das jeweilige Strukturinformationsmittel (350 oder 360) einmal für jeden Chipkartentyp existiert.

Aus dem Attribute-Baum 320 und den Strukturinformationsmitteln 335 werden vorzugsweise zuerst das Anwendungsdatenverzeichnis 390 und das Abbild 370 des zu beschreibenden Speichers der Chipkarte 10 erstellt.

Anwendungsdatenverzeichnis 390

Vorzugsweise wird ein binäres und ein lesbares (z. B. in C-Quellencode) Anwendungsdatenverzeichnis 390, z. B. als Wörterbuch 250, erzeugt. Für jeden Knoten des Typ "Anwendungsdaten" wird ein Eintrag im Wörterbuch 250 sowohl in binärer als auch in lesbarer Form erzeugt. Die Struktur des Eintrags ist durch die interne Struktur-Information 350 vorgegeben. Der Inhalt des Eintrags wird aus dem Attribute-Baum 320 kopiert: Der Name des Eintrags, die Zugriffsart (z. B.: "lies ersten", "lies letzten", ...), die Länge und die Position wird aus dem Knoten des Typ "Anwendungsdaten" kopiert, der Pfad und der Dateityp wird aus dem darüberliegenden Dateiknoten kopiert. Die Knoten des Typ "Anwendungsdaten" können in jeder Reihenfolge verarbeitet werden. Am Ende dieses Vorgangs liegt ein komplettes Anwendungsdatenverzeichnis 390 in binärer Form und eines in lesbarer Form, gemäß der Patentanmeldung P 195 22 527.9, vor. Patentanmeldung P 195 22 527.9 enthält eine detaillierte Beschreibung des Anwendungsdatenverzeichnisses 390.

Abbild 370 (Image, Speicher- oder EEPROM-Abbild)

Vorzugsweise werden die Betriebssystemdaten zuerst an durch die interne Struktur-Informationen 350 definierte Stellen geschrieben. Dann wird der Attribute-Baum 320 in einer vorgegebenen Folge verarbeitet. Aus einem Knoten des Typs "Verzeichnis" wird ein chipkartenspezifisches binäres Abbild des Verzeichnisses für den zu beschreibenden Speicher der Chipkarte 10 (z. B. dem EEPROM) erzeugt. Aus einem Knoten des Typs "Datei" und den darunter liegenden Knoten des Typs "Initialisierungsdaten" und des Typs "Personalisierungsdaten" wird ein chipkartenspezifisches binäres Abbild der Datei für den zu beschreibenden Speicher der Chipkarte 10 erzeugt. Die einzelnen Abbilder werden im Abbild 370 für den zu beschreibenden Speicher der Chipkarte 10 in der Reihenfolge der Verarbeitung abgelegt (Fig. 6). Die Verarbeitung startet vorzugsweise mit dem obersten Verzeichnis im Baum. Dann werden die Unterverzeichnisse des Verzeichnis verarbeitet und danach die Dateien unter dem Verzeichnis. Die Verarbeitung eines Unterverzeichnis erfolgt wie die Verarbeitung des ersten Verzeichnisses, das heißt, der Verarbeitungsprozeß ist rekursiv.

Die Verarbeitung eines Verzeichnis erzeugt aus den Attributen des Verzeichnisknoten und der internen Struktur-Informationen 350 für Verzeichnisse (wie oben für das Anwendungsdatenverzeichnis 390 beschrieben) ein binäres Abbild des Verzeichnisses im zu beschreibenden Speicher. Die Verarbeitung einer Datei erzeugt aus den Attributen der Datei und der darunter liegenden Initialisierungs- und Personalisierungsdatenknoten und der internen Struktur-Informationen 350 für Dateien (wie oben für das Anwendungsdatenverzeichnis 390 beschrieben) ein binäres Abbild der Datei inklusive der Daten der Datei im zu beschreibenden Speicher. Beim Erzeugen der Daten der Datei werden zusätzliche Attribute in die entsprechenden Knoten gesetzt, die auf die erzeugten Daten im binären Abbild 370 des zu beschreibenden Speichers referieren.

Initialisierungstabelle 210

Die Initialisierungstabelle 210 wird aus dem Abbild 370 für den zu beschreibenden Speicher der Chipkarte 10 (z. B. dem EEPROM-Abbild) erzeugt. Die internen allgemeinen Befehls-Informationen 360 der Initialisierungstabelle 380 enthält die Chipkartenkommandos und die (maximalen) Datenlängen. Für jeweils die maximale Datenlänge wird eine Kommandosequenz mit Daten aus dem Abbild 370 erzeugt. Am Anfang der Initialisierungstabelle 210 werden vorzugsweise Kommandosequenzen zur Sicherheitsprüfung eingefügt, und am Ende Kommandosequenzen zum Schreiben des Protokolls und der Statusänderung.

Personalisierungstabelle 220

Die Personalisierungstabelle 220 wird aus dem Attribute-Baum 320 und der internen allgemeinen Befehls-Informationen 360 der Personalisierungstabelle 220 erzeugt. Für jeden Personalisierungsdatenknoten wird eine Kommandosequenz erzeugt. Dabei werden die zusätzlichen Attribute, die bei der Dateierstellung erzeugt wurden, benutzt um die Adressen in dem zu beschreibenden Speicher der Chipkarte 10 zu berechnen. Am Anfang der Personalisierungstabelle 220 werden vorzugsweise Kommandosequenzen zur Sicherheitsprüfung eingefügt, und am Ende Kommandosequenzen zum Schreiben des Protokolls und der Statusänderung.

Kartendefinitionssprache

Die Umsetzung der Beschreibung 310 in die jeweils spezielle interne Struktur 315, wie die Initialisierungsinformationen 350, die Personalisierungsinformationen 360 und die Anwendungsinformationen 340, erfolgt mit Hilfe der speziellen Kartendefinitionssprache. Die Kartendefinitionssprache ist eine deklarative Sprache auf Anwendungsbasis und wird von der Beschreibung 310 angewandt. Als deklarative Sprache wird eine Sprache bezeichnet, die durch Sprachkonstrukte die Datentypen und Strukturen auf denen ein Programm operiert explizit festgelegt wird. Eine deklarative Sprache stellt somit eine Vereinbarung hinsichtlich der Datenstruktu-

ren dar.

Die Kartendefinitionssprache definiert die Elemente der Chipkarte **10**, wie die Verzeichnisse, die Dateien mit den entsprechenden Initialisierungs-Daten, Personalisierungs-Daten und Anwendungs-Daten, sowie die für das Betriebssystem der Chipkarte **10** benötigten Daten. Als Sprach-Elemente können Anweisungen, Namen, Werte, Symbolische Werte, Ausdrücke und Kommentare verwendet werden.

Anweisungen stellen entweder konstruktive Anweisungen, die die Elemente der Chipkarte **10** definieren, oder Attribut-Anweisungen, die Eigenschaften der Elemente beschreiben, dar.

Attribut-Anweisungen definieren Eigenschaften von Elementen und sind in konstruktiven Anweisungen enthalten und sind optional. Innerhalb einer konstruktiven Anweisung können Attribut-Anweisungen in jeder Reihenfolge spezifiziert werden. Die gleiche Eigenschaft kann mehrmals spezifiziert werden. Die letzte Anweisung ist gültig. Anweisungen werden vorzugsweise definiert als: ANWEISUNG oder ANWEISUNG (element1, element2,...), wobei element1, element2,... für Anweisungen, Namen, Werte, symbolische Werte oder Ausdrükke stehen können.

Namen werden in konstruktiven Anweisungen benutzt, um die erzeugten Elemente der Chipkarte **10** zu identifizieren. Namen sind optional, wenn kein Name spezifiziert ist, wird ein interner Name erzeugt.

Werte sind Zeichenketten denen durch Auswertung ein arithmetischer Wert zugewiesen werden kann. Zeichenketten können Längen haben von 0 bis zur maximalen Größe des zu beschreibenden Speichers, z. B. eines EEPROM.

Symbolische Werte sind Symbole, die einen Wert darstellen. Die symbolische Werte werden in einer Deklarationsanweisung definiert, z. B.:

```
Deklaration
(
Symbol1 (Ausdruck)
Symbol2 (Ausdruck)
...
).
```

Symbolische Werte können auch innerhalb eines Ausdrucks zugewiesen werden mit, z. B. mit einer Zuweisung durch den Operator "=". Symbolische Werte können in späteren Deklarationen benutzt werden. In dem obigen Beispiel, könnte Symbol1 in dem Ausdruck für Symbol2 benutzt werden. Symbole können mehr als einmal deklariert werden. Spätere Deklarationen überschreiben vorhergehende.

Werte können z. B. dargestellt werden mit arithmetischen Ausdrücken, boolschen Ausdrücken, Zeichenketten-Ausdrücken, symbolischen Ausdrücken und Zuweisungsausdrücken. Vorzeichen, wie "+" und "−" können mit einer bestimmten vorgebbaren Darstellungsweise, beispielsweise in Klammern, z. B. (−122), verwendet werden.

Ausdrücke werden vorzugsweise von links nach rechts verarbeitet. Eine Operatoren-Priorität kann z. B. mit Klammern gesteuert werden. Anweisungen und Ausdrücke können sich über mehrere Zeilen erstrecken. Text-Zeichenketten müssen in einer Zeile enthalten sein. Mit einem Anfügungsoperator, z. B. ",", können längere Zeichenketten erzeugt werden.

Kommentare können zur Verbesserung der Lesbarkeit und der Verständlichkeit eingefügt werden.

Layout Definition

Die Strukturbeschreibung **310** (auch Layout Definition genannt) einer Chipkarte **10** besteht vorzugsweise aus vier Sektionen, dem Prolog, den Betriebssystemdaten der Chipkarte **10**, dem Verzeichnis/Datei Baum und den Protokolldaten.

Der Prolog ist optional und besteht aus Deklarations- und/oder Umgebungsanweisungen. Eine Deklarations-Anweisung definiert symbolische Werte. Diese symbolischen Werte können später in Ausdrücken benutzt werden. Eine Umgebungsanweisung spezifiziert den Typ der Chipkarte **10** und Parameter für die Personalisierungsinformation **360** und die Initialisierungsinformation **350**.

Das folgende Beispiel eines Prologs definiert unter anderem die Speichergröße des verfügbaren Speichers, dessen Anfangsadresse, und dem Betriebssystemtyp der Chipkarte **10**.

```
Environment          /* Beginn der Prolog Deklarationen */
(
   Eeprom_Size ( 8128 )              /* Speichergröße   */
   Start_Address ( 0x0100 )          /* Startadresse    */
   ROM_Mask_Id ( "MFC Version 2.0" ) /* Betriebssystem */
)
```

Die Betriebssystemdaten definieren Initialwerte für Bereiche des beschreibbaren Speichers auf der Chipkarte **10**, die zur Erweiterung oder Veränderung des Betriebssystems der Chipkarte **10** vorgesehen sind. Sie bestehen

aus den Deklarationsanweisungen für diese Initialwerte. Beispiel:

```
Declarations
(
  Jump ( 0xCC )          /* Sprunginstruktion                    */
  AltReset ( 0x0000 )    /* Alternate reset address              */
  EndProcess ( 0x7FF0 ) /* End Process address                  */

  ATR_Table (            /* Tabelle mit Daten für den ATR        */
              0x3B,      /* direct convention                    */
              0x86,      /* T0  : 6 historical chars             */
              0x81,      /* TD1 : TD2 follows                    */
              0x31,      /* TD2 : TA3, TB3 follows               */
              0x40,      /* TA3 : T=1 ICC Info field size        */
              0x34,      /* TB3 : T=1 Block/Char waiting time    */
              "IBMMFC", /* Historical characters                */
              0x05       /* CRC */
            )
  ...
)

Fixed_Data
(
   0, 0, 0, 0,           /* 4 byte reservierter Speicher         */
   Jump, EndProcess,     /* Sprungtabelle für Veränderungen      */
   Jump, AltReset,       /* Sprungtabelle für Veränderungen      */
   Sizeof(ATR_Table),    /* Länge der ATR Tabelle                */
   ATR_Table             /* ATR Tabelle wird hier eingefügt      */
)
```

Der Attribute-Baum 320 wird mit einer Verzeichnis/Datei Anweisung definiert. Den Verzeichnissen und Dateien werden symbolische Namen zugeordnet. Eine Verzeichnis/Datei-Eigenschaft Anweisung definiert das Kennzeichen des Verzeichnis, die Zugriffs-Eigenschaften und Sicherheitsbereiche, z. B. für die Generierung des Anwendungsdatenverzeichnisses 390. Sie können in jeder Reihenfolge und mehrmals definiert werden.

Initialisierungsdaten- und Personalisierungsdaten-Anweisungen definieren die Initialisierungs- und Personalisierung-Daten für eine Datei. Die Initialisierungs-Daten Anweisung definiert Daten in der Datei, die zum Zeitpunkt der Initialisierung geschrieben werden.

Die Personalisierungsdaten-Anweisung definiert Felder in der Datei die später individuell für jede Chipkarte 10 personalisiert werden. Sie definiert die Länge eines Personalisierungs-Daten-Feld, einen symbolischen Index, welcher das Daten Feld mit dem Personalisierungs-Datensatz korreliert, und eine optionale Sicherheitsmethode.

Eine Anwendungsdaten-Anweisung definiert Daten-Felder für die Anwendungsprogramme. Die Daten-Felder erzeugen Einträge, z. B. in dem Anwendungsdatenverzeichnisses 390.

Zugriffs-Eigenschaften werden definiert mit einer Zugriffs-Anweisung. Die Zugriffs-Eigenschaften definieren sicherheitsrelevante Eigenschaften und die verwendbaren Zugriffsmethoden für die Daten.

Das folgende Beispiel zeigt die Beschreibung einer Verzeichnisstruktur, wie zum Beispiel der aus Fig. 2, in einer Kartendefinitionssprache:

```
Directory          /* Deklaration eines Verzeichnisses      */
( DIR_100          /* Symbolischer Name des Verzeichnisses */
   ...
   Directory       /* Deklaration eines Verzeichnisses      */
   ( DIR_120       /* Name des Unterverzeichnisses          */
      ...
      File         /* Deklaration einer Datei               */
      ( FILE130    /* Symbolischer Name der Datei           */
         ...
         Access    /* Definition der Zugriffseigenschaften */
         (  Read ( ALWAYS )
            Update ( PROTECTED )
         )
         /* Daten für das Anwendungsdatenverzeichniss     */
         Agent_Data ( CARD_SEQUENCE_NUMBER length(10) )
         /* Daten für die Initialisiserungstabelle        */
         Initial_Data ( 999999999 )
         /* Daten für die Personalisierungstabelle        */
         Personal_Data ( ItemNo123 )
         ...
      )

      File         /* Deklaration einer Datei               */
      ( FILE140    /* Symbolischer Name der Datei           */
         ...
         /* Daten für das Anwendungsdatenverzeichniss     */
         Agent_Data ( CARD_COUNTRY_CODE    length( 2) )
         /* Daten für die Initialisiserungstabelle        */
         Initial_Data ( "DE" )
         ...      /* weitere Daten */
      )
   )
   Directory       /* Deklaration eines Verzeichnisses      */
   ( DIR_150       /* Name des Unterverzeichnisses          */
      ...
   )
)
```

Im obigen Beispiel werden die Verzeichnisse 100, 120 und 150 angelegt. Das Verzeichnis 120 ist ein Unterverzeichnis von Verzeichnis 100 und enthält selbst wiederum die Dateien 130 und 140. In Datei 130 wurde eine Kartensequenznummer mit 10 Stellen abgelegt. Der Initialwert ist 999999999 und wird bei der Personalisierung

als "ItemNo123" durch den aktuellen Wert ersetzt. Die Zugriffseigenschaften der Datei erlauben einen uneingeschränkten Lese-Zugriff aber einen Schreibzugriff nur nach entsprechender Sicherheitsprüfung. Die Datei 140 enthält einen Ländercode von 2 Stellen Länge und wird mit dem Text "DE" vorbelegt.

Der Typ der Chipkarte 10 wird vorzugsweise definiert mit Angabe über den Prozessortyp der Chipkarte, der Speichergröße des beschreibbaren Speichers, z. B. einem EEPROM, sowie dessen Anfangsadresse, und dem Betriebssystemtyp der Chipkarte.

Patentansprüche

1. Vorrichtung (300) zur Erstellung eines Werkzeuges (315) für eine Modifikation von Daten auf einer Chipkarte (10) oder für eine Kommunikation mit der Chipkarte (10), wobei die Vorrichtung (300) als Eingangsdaten eine Beschreibung (310) in einer hierfür vorgesehenen Kartendefinitionssprache erhält und die Beschreibung (310) Information über Objekte auf der Chipkarte (10), die mit Hilfe konstruktiver Anweisungen beschrieben sind, aufweist; mit:
einem Mittel zur Generierung eines internen Attributeverzeichnisses (320) aus den Informationen über die Objekte in der Beschreibung (310), wobei für jedes Objekt, mindestens ein Eintrag (330) in dem internen Attributeverzeichnis (320) existiert, und
zumindest einem Strukturinformationsmittel (335), das beschreibt, wie die in dem mindestens einen Eintrag (330) in der internen Attributeverzeichnis (320) gespeicherten Daten in das für die Chipkarte (10) bestimmte Format gebracht werden, welche Informationen aus dem mindestens einen Eintrag (330) verwendet und in welchem Format sie in der speziellen Struktur abgelegt werden.

2. Verfahren zur Erstellung eines Werkzeuges (315) für eine Modifikation von Daten auf einer Chipkarte (10) oder für eine Kommunikation mit der Chipkarte (10), mit den folgenden Schritten:
Eingabe einer Beschreibung (310) der Chipkarte (10) in einer hierfür vorgesehenen Kartendefinitionssprache, wobei die Beschreibung (310) Information über Objekte auf der Chipkarte (10), die mit Hilfe konstruktiver Anweisungen beschrieben sind, aufweist;
Generierung eines internen Attributeverzeichnisses (320) aus den Informationen über die Objekte in der Beschreibung (310), wobei für jedes Objekt mindestens ein Eintrag (330) in dem internen Attributeverzeichnis (320) existiert,
Eingabe und/oder Abruf zumindest eines Strukturinformationsmittels (335), das beschreibt, wie die in dem mindestens einen Eintrag (330) in dem internen Attributeverzeichnis (320) gespeicherten Daten in das für die Chipkarte (10) bestimmte Format gebracht werden sollen, welche Informationen aus dem mindestens einen Eintrag (330) verwendet werden und in welchem Format sie in dem Werkzeug (315) abgelegt werden, und
Erzeugung des Werkzeuges (315) aus der Beschreibung (310) mit Hilfe des internen Attributeverzeichnisses (320) und des zumindest einen Strukturinformationsmittels (335).

3. Vorrichtung und/oder Verfahren nach einem der vorstehenden Ansprüche, dadurch gekennzeichnet, daß das Werkzeug (315) ein Anwendungsdatenverzeichnis (390) ist, wobei das Anwendungsdatenverzeichnis (390) anwendungsspezifische Daten die für eine Kommunikation mit der Chipkarte erforderlich sind, aufweist.

4. Vorrichtung und/oder Verfahren nach Anspruch 3, dadurch gekennzeichnet, daß das Anwendungsdatenverzeichnis (390) ein Wörterbuch (250) gemäß der Patentanmeldung P 19522527.9 ist.

5. Vorrichtung und/oder Verfahren nach einem der vorstehenden Ansprüche, dadurch gekennzeichnet, daß das Werkzeug (315) ein Mittel für eine Initialisierung und/oder Personalisierung der Chipkarte (10) ist.

6. Vorrichtung und/oder Verfahren nach einem der vorstehenden Ansprüche, dadurch gekennzeichnet, daß die Modifikation von Daten auf der Chipkarte (10) eine Bereitstellung der Chipkarte (10) durch Initialisierung und/oder Personalisierung der Chipkarte (10) darstellt.

7. Vorrichtung und/oder Verfahren nach Anspruch 6, dadurch gekennzeichnet, daß das Werkzeug (335) ein Abbild (370) eines zu beschreibenden Speichers der Chipkarte (10) oder eine Tabelle (380) für die Initialisierung und/oder Personalisierung der Chipkarte (10) ist.

8. Vorrichtung und/oder Verfahren nach einem der vorstehenden Ansprüche, dadurch gekennzeichnet, daß die Beschreibung (310) Informationen über Betriebssystemdaten, Verzeichnisse, Dateien, Initialisierungsdaten und Personalisierungsdaten der Chipkarte (10), und/oder Anwendungsdaten für eine Kommunikation mit der Chipkarte (10) aufweist.

9. Vorrichtung und/oder Verfahren nach einem der vorstehenden Ansprüche, dadurch gekennzeichnet, daß die Beschreibung (310) Attribute-Anweisungen aufweist, die die Eigenschaften der Objekte spezifizieren.

10. Vorrichtung und/oder Verfahren nach einem der vorstehenden Ansprüche, dadurch gekennzeichnet, daß die auf der Chipkarte (10) erreichbaren Daten in der Beschreibung (310) mit dafür vorgesehenen Alias-Namen versehen sind.

11. Vorrichtung und/oder Verfahren nach einem der vorstehenden Ansprüche, dadurch gekennzeichnet, daß die Beschreibung (310) in einer für Menschen lesbaren Form vorliegt.

12. Vorrichtung und/oder Verfahren nach einem der vorstehenden Ansprüche, dadurch gekennzeichnet, daß der mindestens eine Eintrag (330) einen Namen des Objektes, einen Typ des Objektes, Attribute des Objektes und Verknüpfungen mit anderen Einträgen im Attributeverzeichnis (320) aufweist.

13. Vorrichtung und/oder Verfahren nach einem der vorstehenden Ansprüche, dadurch gekennzeichnet, daß das Attributeverzeichnis (320) unabhängig von einem jeweiligen, verwendeten Typ der Chipkarte (10) ist.

14. Vorrichtung und/oder Verfahren nach einem der vorstehenden Ansprüche, dadurch gekennzeichnet,

daß das Strukturinformationsmittel (335) Anwendungsdaten-Informationen (340), Struktur-Informationen (350) und/oder Befehls-Informationen (360) aufweist.

15. Vorrichtung und/oder Verfahren nach Anspruch 14, dadurch gekennzeichnet, daß die Anwendungsdaten-Informationen (340) ein allgemeines Layout des Werkzeuges (315) aufweisen.

16. Vorrichtung und/oder Verfahren nach Anspruch 14 oder 15, dadurch gekennzeichnet, daß die Struktur-Informationen (350) Informationen über ein Verzeichnis als allgemeines Layout der internen physischen Struktur auf der Chipkarte (10), über Dateien als allgemeines Layout der internen physischen Struktur auf der Chipkarte (10), und über Betriebssystemdaten als allgemeines Layout der internen Betriebssystemstruktur, im entsprechenden Format der Chipkarte (10) aufweisen.

17. Vorrichtung und/oder Verfahren nach Anspruch 14—16 wenn in Kombination mit Anspruch 7, dadurch gekennzeichnet, daß die Befehls-Informationen (360) Informationen über die Initialisierungstabelle (210) und die Personalisierungstabelle (220) als allgemeines Layout der jeweiligen Tabelle und der entsprechenden erforderlichen Kommandos, aufweisen.

18. Vorrichtung und/oder Verfahren nach einem der vorstehenden Ansprüche, dadurch gekennzeichnet, daß das Attributeverzeichnis (320) ein sequentielles hierarchisches Verzeichnis ist, das vorzugsweise eine Baumstruktur bildet.

Hierzu 5 Seite(n) Zeichnungen

Nummer: DE 195 36 548 A1
Int. Cl.[6]: G 06 F 9/45
Offenlegungstag: 3. April 1997

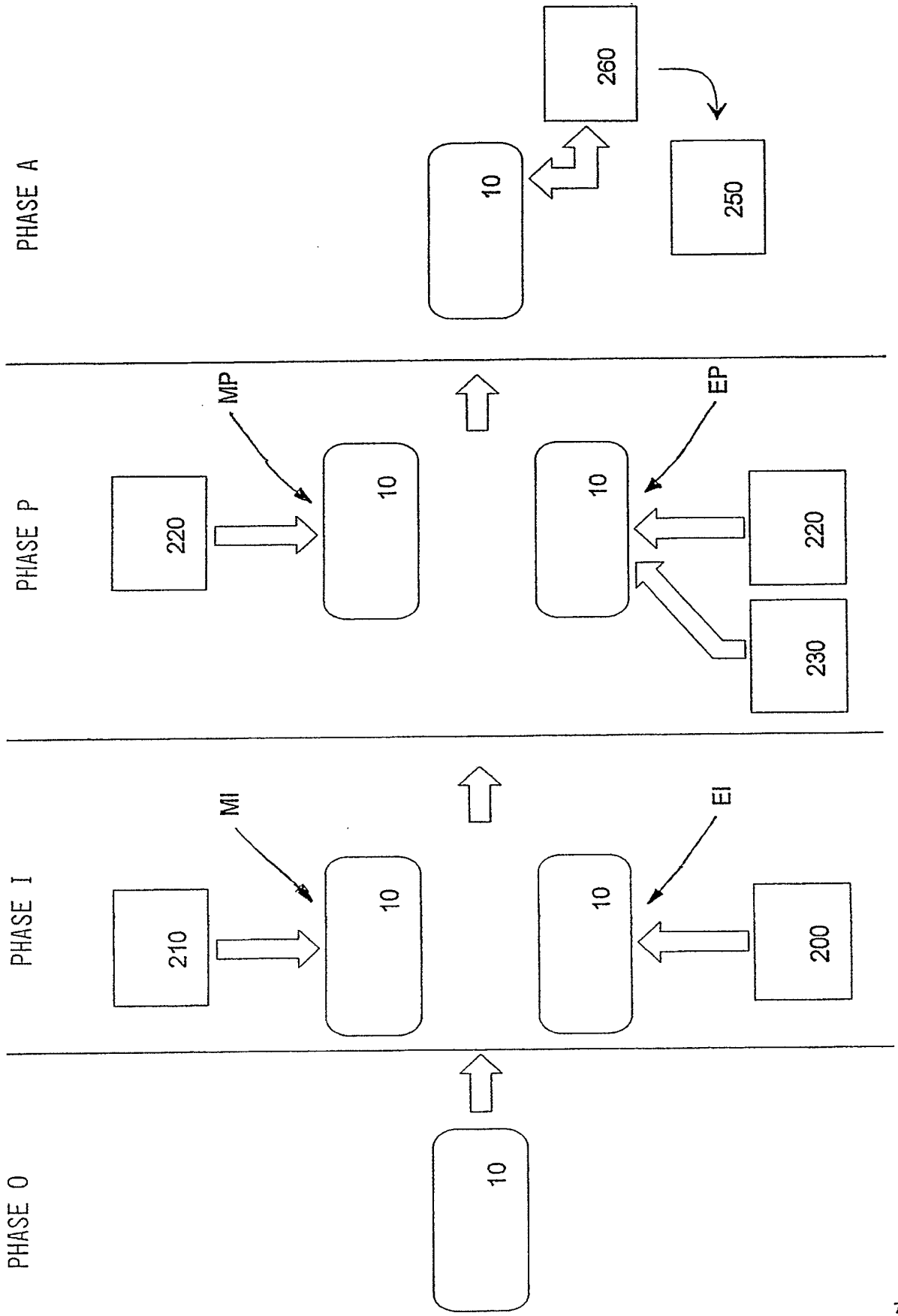


FIG. 1

Nummer: DE 195 36 548 A1
Int. Cl.[6]: G 06 F 9/45
Offenlegungstag: 3. April 1997

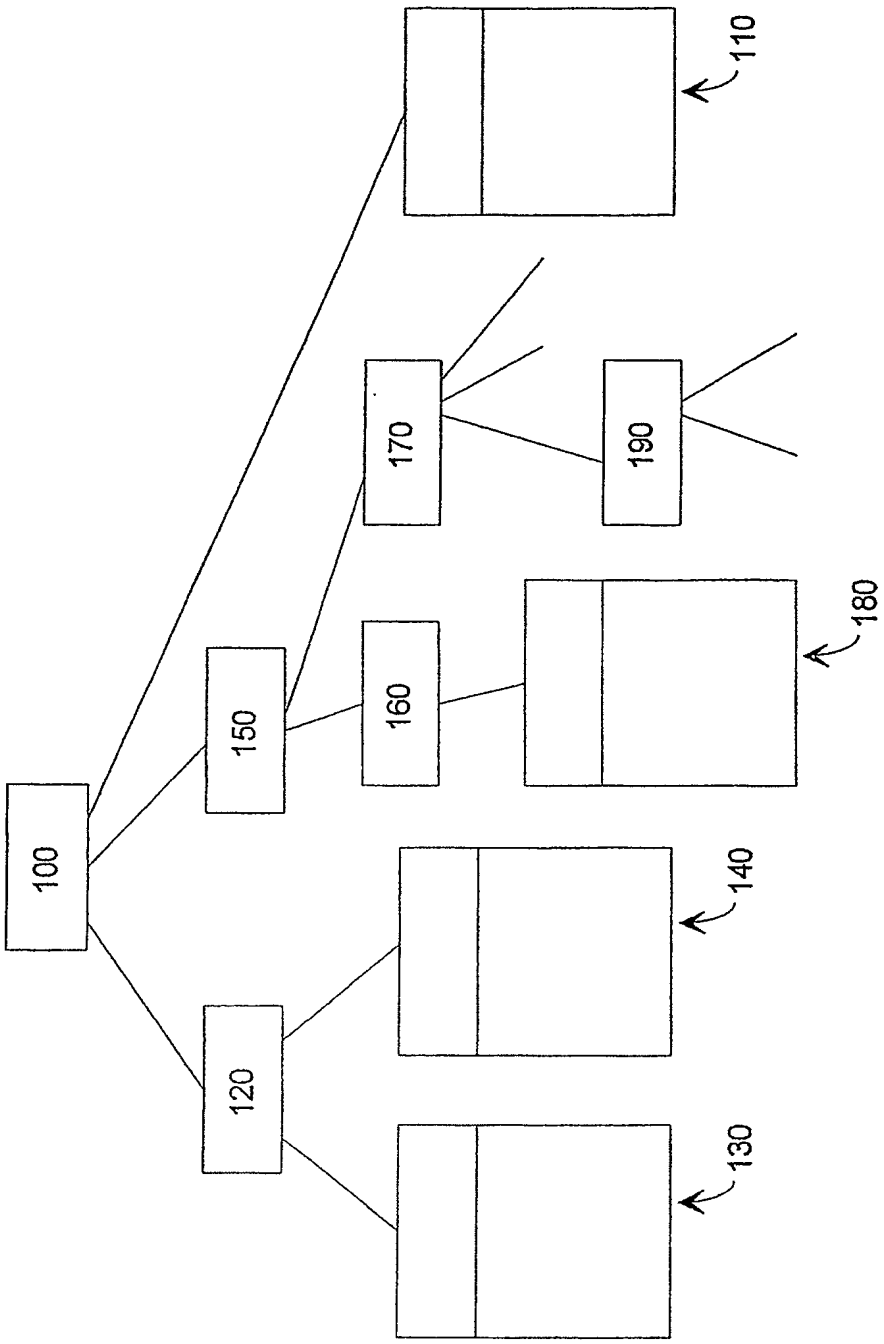


FIG. 2

Nummer: DE 195 36 548 A1
Int. Cl.[6]: G 06 F 9/45
Offenlegungstag: 3. April 1997

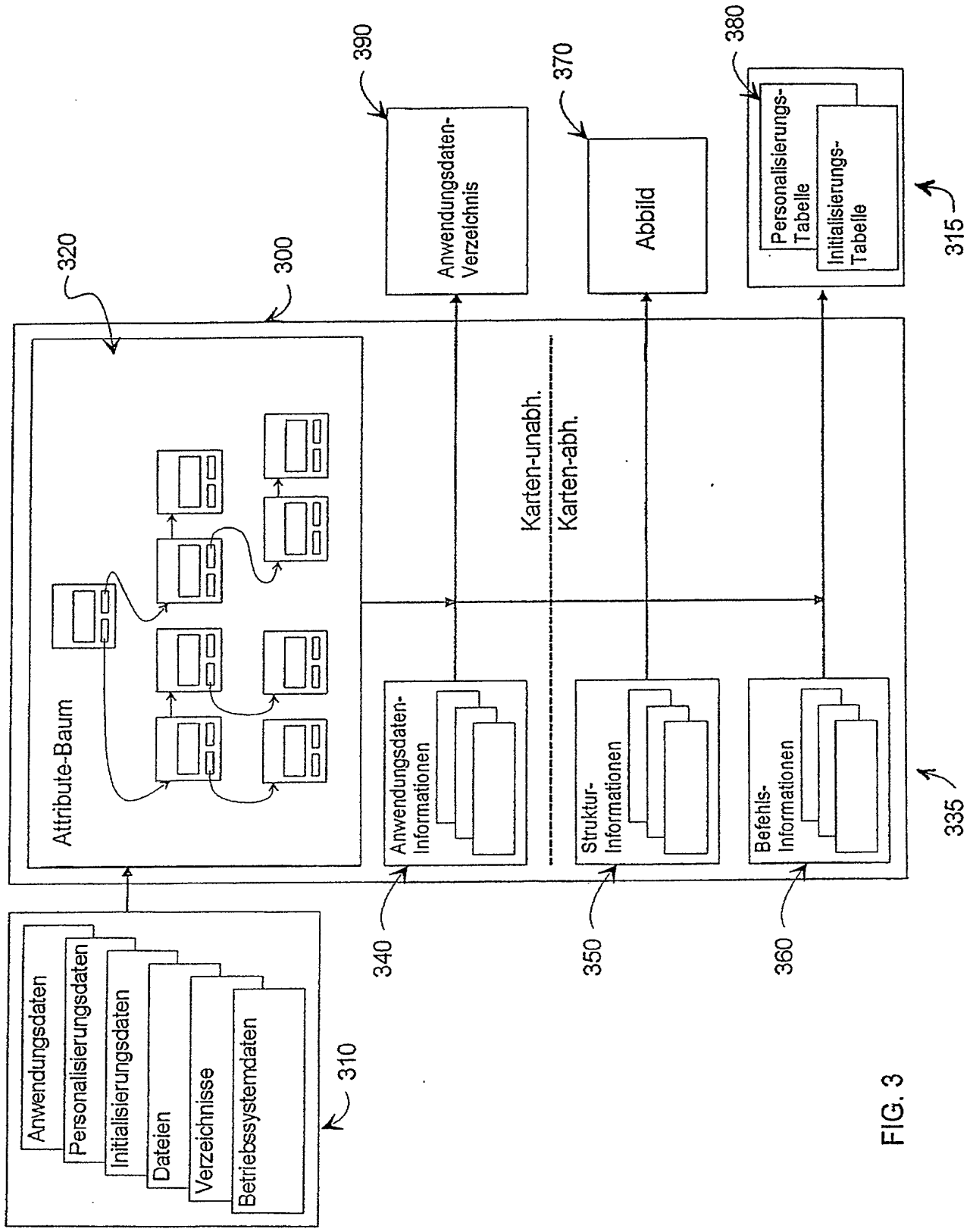


FIG. 3

Nummer: DE 195 36 548 A1
Int. Cl.[6]: G 06 F 9/45
Offenlegungstag: 3. April 1997

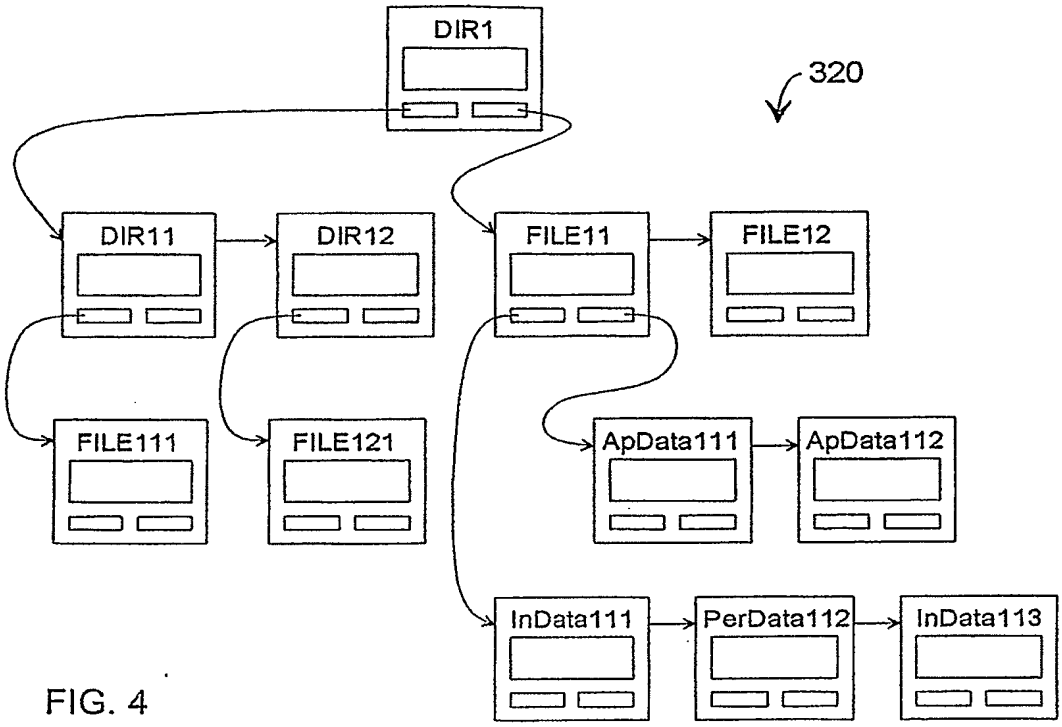


FIG. 4

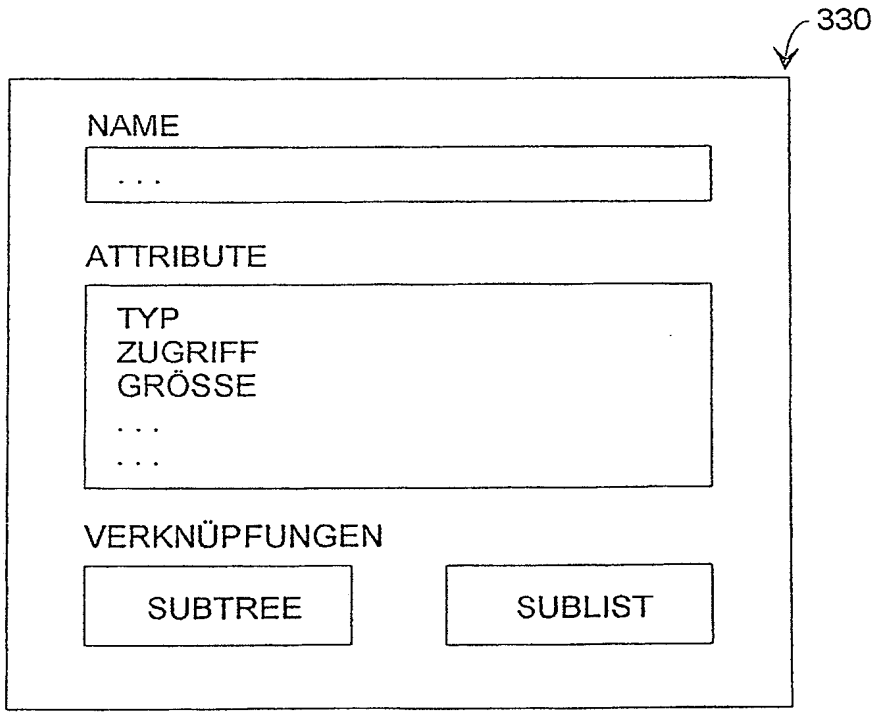


FIG. 5

Nummer: DE 195 36 548 A1
Int. Cl.[6]: G 06 F 9/45
Offenlegungstag: 3. April 1997

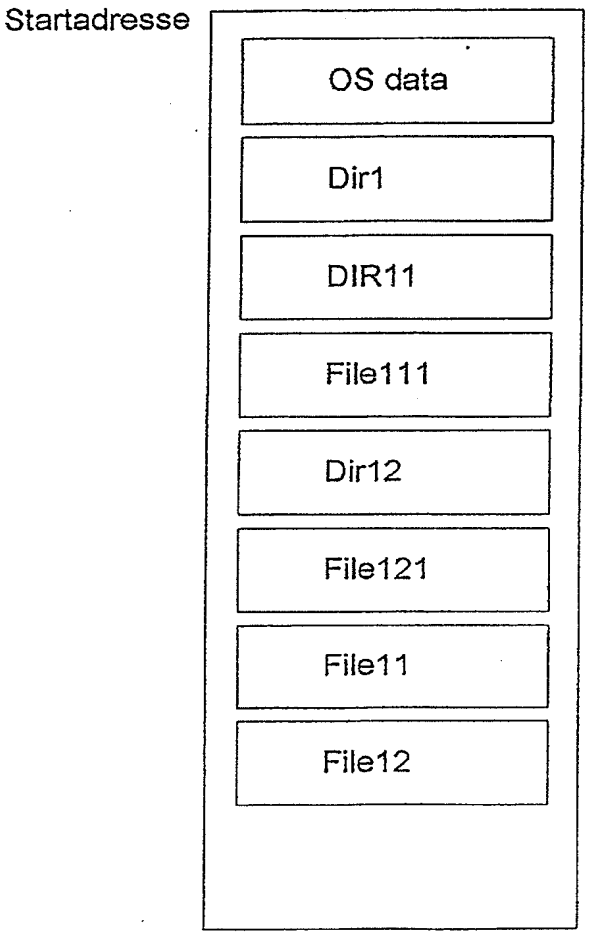


FIG. 6